

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-149048

(43)Date of publication of application : 30.05.2000

(51)Int.Cl.

G06T 15/00

(21)Application number : 11-292817

(71)Applicant : MITSUBISHI ELECTRIC INF
TECHNOL CENTER AMERICA INC

(22)Date of filing : 14.10.1999

(72)Inventor : KNITTEL JAMES
PEET WILLIAM
CORRELL KENNETH

(30)Priority

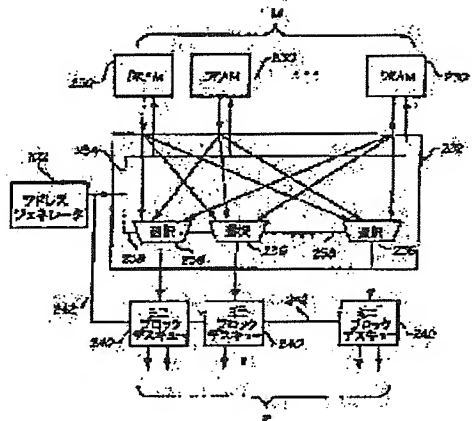
Priority number : 98 191865 Priority date : 12.11.1998 Priority country : US

(54) REAL-TIME VOLUME RENDERING ATTAINING METHOD

(57)Abstract:

PROBLEM TO BE SOLVED: To change fundamental architecture for the storage and distribution of voxels, in order to attain the performance of real-time volume rendering.

SOLUTION: Voxel data are stored in mini blocks which are allocated to a set of DRAM modules in a volume rendering system. Then a double-layered skewing architecture is assembled in a memory subsystem of the volume rendering system. Thus, the data can be transferred at the highest burst rate for a DRAM, and real-time volume rendering is attained. In each DRAM module, the mini blocks are allocated to the memory banks so that the miniblocks which are continuously accessed are allocated to the different banks so as to avoid idle cycles during the transfer of data and to cause the DRAM transfer efficiency to improve up to almost 100%.



(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-149048

(P2000-149048A)

(43) 公開日 平成12年5月30日 (2000.5.30)

(51) IntCl.
G 0 6 T 15/00

識別記号

F I
G 0 6 F 15/72

テマコード (参考)

4 5 0 K

審査請求 有 請求項の数 7 O L (全 19 頁)

(21) 出願番号 特願平11-292817
(22) 出願日 平成11年10月14日 (1999.10.14)
(31) 優先権主張番号 09/191865
(32) 優先日 平成10年11月12日 (1998.11.12)
(33) 優先権主張国 米国 (US)

(71) 出願人 597067574
ミツビシ・エレクトリック・インフォメイ
ション・テクノロジー・センター・アメリ
カ・インコーポレイテッド
MITSUBISHI ELECTRIC
INFORMATION TECHNO
LOGY CENTER AMERIC
A, INC.
アメリカ合衆国、マサチューセッツ州、ケ
ンブリッジ、ブロードウェイ 201
(74) 代理人 100057874
弁理士 曾我 道照 (外6名)

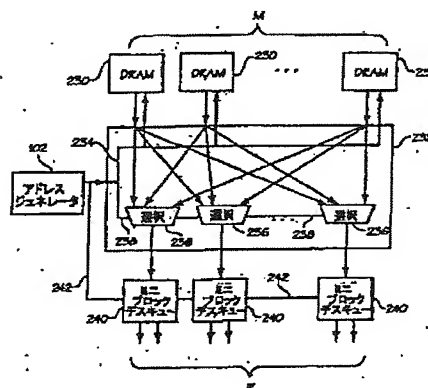
最終頁に続く

(54) 【発明の名称】 実時間ボリュームレンダリングを可能にする方法

(57) 【要約】

【課題】 実時間ボリュームレンダリングのパフォーマンスを達成するために、ボクセルの記憶および分散の根本的なアーキテクチャにおける変更を提供する。

【解決手段】 1組のDRAMメモリモジュールに割り当てられたミニブロック内にボクセルデータが記憶されたボリュームレンダリングシステムのメモリサブシステム上に2層のスキューイングアーキテクチャを組み付け、それによって、DRAMメモリの最大バーストレートでのデータ転送を可能にし、実時間ボリュームレンダリングを可能にする。各DRAMモジュール内では、連続してアクセスされるミニブロックが異なるバンクに割り当てられるように、ミニブロックがメモリバンクに割り当てられ、それによってデータ転送中のアイドルサイクルを回避し、DRAM転送効率をほぼ100%に増加させる。



【特許請求の範囲】

【請求項1】 ボクセルデータをDRAMメモリモジュールに配置して、ボリュームレンダリングシステムにおけるデータ転送中のアイドルサイクルを回避し、それによってDRAMメモリモジュールの最大バーストレートでのデータ転送による実時間ボリュームレンダリングを可能にする方法であって、

ボクセルデータ集合からのボクセルデータをミニブロックに配置するステップと、

それぞれ多数のDRAMメモリバンクを有する多数のDRAMモジュールを提供するステップと、

各DRAMモジュール内で、連続的にアクセスされるミニブロックが関連するDRAMモジュール内で異なったバンクに割り当てられるように、ミニブロックをそのメモリバンクに割り当てるステップであり、上記ミニブロックは、特定のDRAMと2層記憶システムにおけるDRAM内の特定のバンクとの両方に割り当てられ、上記バンクの連続的アクセスがあった時点で、関連するDRAMモジュールからのデータ転送の間のアイドルサイクルが回避されるステップとを含む実時間ボリュームレンダリングを可能にする方法。

【請求項2】 所定の順序でDRAMメモリのバンクからボクセルデータを読み出すステップと、上記読み出しがコンフリクトにつながる場合を検出するステップと、上記コンフリクトの検出に応じて読み出し順序を反転させるステップとをさらに含むことを特徴とする請求項1に記載の実時間ボリュームレンダリングを可能にする方法。

【請求項3】 上記コンフリクトは、現在読み出されているのと同じメモリバンクからの連続的読み出しを含むことを特徴とする請求項2に記載の実時間ボリュームレンダリングを可能にする方法。

【請求項4】 DRAMメモリから読み出されたボクセルデータを再度順序付けして、上記ボクセルが上記ボリュームデータ集合に配置された順序でのボクセル処理を可能にするステップをさらに含むことを特徴とする請求項1に記載の実時間ボリュームレンダリングを可能にする方法。

【請求項5】 上記ミニブロックのそれぞれは、 $2 \times 2 \times 2$ ボクセルの立体配列での8ボクセルを含むことを特徴とする請求項1に記載の実時間ボリュームレンダリングを可能にする方法。

【請求項6】 上記ミニブロックは、次式に従って上記DRAMモジュールに記憶され、

$$\text{DRAM Module Number} = (X_{mb} + Y_{mb} + Z_{mb}) \bmod M$$

ここで、 $X_{mb} + Y_{mb} + Z_{mb}$ は次式によって求められる座標(X, Y, Z)を有するボクセルを含むミニブロックの位置を定義する。

$$X_{mb} = [X/2], \quad Y_{mb} = [Y/2], \quad Z_{mb} = [Z/2]$$

M個のモジュールの全ては、1つのモジュールにアクセスするのに必要な時間と同じ長さで同時にアクセスでき、ミニブロックの座標を合計し、Mで割って剰余を取ることによって、上記ボリュームデータ集合の何れかの軸と同一線上のM個のブロックの何れかのグループが一斉かつ同時にフェッチされることを保証することを特徴とする請求項1に記載の実時間ボリュームレンダリングを可能にする方法。

【請求項7】 座標(X_{mb} , Y_{mb} , Z_{mb})を有するミニブロックは、次式に従ってバンクに割り当てられる

$$\text{Bank Number} = [(X_{mb}/M) + (Y_{mb}/M) + Z_{mb}/M] \bmod B$$

ここで、Mは独立のメモリチップの数であり、Bは1チップ当たりのバンクの数であり、また、BはDRAMチップ1個当たりのバンクの数である。ことを特徴とする請求項6に記載の実時間ボリュームレンダリングを可能にする方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】この発明は、実時間ボリュームレンダリングを可能にする方法に関し、ボリュームレンダリング、より詳細には、ボリュームデータ集合の記憶に最小サイズのデータブロックを用いてメモリの迅速な読み出しを介して実時間ボリュームレンダリングを可能にするメモリアーキテクチャに関する。

【0002】

【従来の技術】ボリュームレンダリングは、3以上の次元におけるサンプルデータとして表すオブジェクトまたは事象の視覚化を扱うコンピュータグラフィックスの分野である。ボリュームグラフィックスの一部である。これらのサンプルはボリュームエレメント、すなわち「ボクセル」と呼ばれ、検討されるオブジェクトや事象の物理的特徴を表すデジタル情報を含む。ボリュームレンダリングは、印刷、コンピュータ端末での表示および他の形態の視覚化を目的として、二次元画像としてのボリュームデータの投影に関するボリュームグラフィックスの一部である。実時間ボリュームレンダリングは、通常は1秒間に30フレーム以上の、高速で連続する一連の画像としてのボリュームデータの投影および表示であり、これによって、オペレータがユーザーに視覚的なフィードバックを逐一提供しながら投影のパラメータを対話的に制御し、画像を操作することを可能にする。

【0003】ボリュームレンダリング用のソフトウェア手法は10年から20年に亘って実施されてきたが、要求される膨大な計算能力と、十分な速度でボクセルデータを読み出し・移動することの困難さとの両方の理由から、実時間ボリュームレンダリングには適していなかった。現代のパーソナルコンピュータの急速な能力向上をもってしても、今後何年にも亘ってソフトウェア上で実時間ボリュームレンダリングを支援するには十分とはな

りそうもない。例えば、各エッジに256ボクセルで、合計で256³または1600万ボクセルあるポリウムデータ集合を表現し、これを実時間で行うには、1秒当たり30回以上で全ての1600万ボクセルを読み出し処理することが必要である。これは1秒間に5億ボクセルを超えるレートで読み出しおよび処理をしなければならないことになり、このレートは現代のパーソナルコンピュータで利用できる計算能力およびメモリ帯域幅を遥かに超えるものである。512³ボクセルのポリウムデータ集合は、その8倍の(読み出しおよび処理)レート、すなわち1秒間に約40億ボクセルのレートを必要とし、1024³ボクセルのポリウムデータ集合はさらにその8倍の(読み出しおよび処理)レート、すなわち1秒間に約320億ボクセルのレートを必要とすることがわかる。各フレームにおいて処理されるボクセルの数を削減するために確立されたソフトウェア技術を使用したとしても、レートは依然として現代のパーソナルコンピュータのメモリ帯域幅および計算能力を超える。

【0004】しかし、現代の半導体技術を用いれば、特殊な目的のためのポリウムレンダリングシステムを、例えば回線基盤にプラグインで取り付けるパーソナルコンピュータの付属品として構築することが可能になった。かかるシステムにおいては、ボクセルはDRAMチップとも呼ばれる複数のダイナミックランダムアクセスメモリのモジュールに記憶される。データは1つ以上の平行なパイプライン化された処理エレメントによって読み取りおよび処理され、実時間のフレームレートで画像を投影する。かかる特殊なシステムでは、いかにして十分な速度でメモリからボクセルデータを読み出すかが難問となる。この速度は、バーストモード、すなわち隣接したメモリアドレスに記憶された一連のデータ値を連続的にすばやく読み出すモード、で動作する最速のDRAMチップを除けば、すべてのメモリの帯域幅を超えている。バーストモードのDRAMチップであったとしても、メモリの効率をほぼ100%まで最大化し、定格帯域幅のほぼ100%でチップを動作させることが必要となる。

【0005】1997年8月1日に出願されており、本明細書中に参考文献として含めた米国特許出願第08/905,238号において、実時間ポリウムレンダリングシステムは、一つのブロック内のすべてのボクセルが単一のメモリモジュール内の隣接したメモリアドレスに逐次格納されるように、ボクセルデータがブロックに編成されている、と説明されている。これによって、一度に一つというよりも、ブロック全体のデータを一気にフェッチすることが可能となり、それによってDRAMに関連するバーストモードのアクセスを利用することができる。ボクセルのブロックがフェッチされると、ボクセルはパイプライン毎に1サイクル当たり1ボクセルのレートで、1つ以上の処理パイプラインに送られる。そ

の間に、それに続くボクセルのブロックのフェッチングが始まる。典型的な高性能DRAMチップは、それぞれ7.5ナノ秒、7ナノ秒および6ナノ秒のサイクル時間に対応して、1秒間に1億3300万、1億4700万または1億6600万のレートで動作させることができる。各ポリウムの値がそれぞれ1つのDRAMデータエレメントからなるとすると、1秒間に5億ボクセルという必要データレートを達成するために、およそ4つのDRAMチップを並行して動作させる必要がある。

【0006】ボクセルデータのブロックを読み出す順序は、ポリウムデータ集合を見る方向、すなわちポリウムデータ集合に対する画像平面の位置によることがお分かり戴けるであろう。何れの見方についても必要なボクセル読み出しおよび処理レートを達成するため、DRAMチップの並行操作にコンフリクトがないように、実時間ポリウムレンダリングシステムのDRAMチップ全体に渡ってボクセルデータを分散することが必要である。これは、Cube-4というシステムで実施されており、1996年12月にHanspeter PfisterがStony Brookのニューヨーク州立大学のコンピュータサイエンス学部に提出した「Architectures for real-time Volume Rendering」という題の博士論文に記載され、さらに米国特許第5,594,842号「実時間ポリウム視覚化のための装置と方法」に記載されたボクセルデータを「スキューイング:skewing」する方法によって達成される。このスキューイングの方法は改良され、米国特許出願第08/905,238号に記載された、EM-Cubeというシステムにおけるボクセルのブロックのメモリ組織に適合された。

【0007】Cube-4システムのスキューイングの本質は、隣接するボクセルが異なったDRAMチップに記憶されているということである。これは3つの全ての次元にあてはまり、従って同じ数のDRAMチップから、ポリウムデータ集合の何れかの軸と一線上に並べられた隣接するボクセルの何れかのグループを同時にフェッチすることが可能である。これによってDRAMチップを並行に使用する効率は最大化されるが、各DRAMチップの帯域幅を非効率的に利用することになる。EM-Cubeシステムの本質は、個々のボクセルではなくて隣接するボクセルのブロックが、隣接するDRAMチップに記憶されることである。これによって各DRAMチップの帯域幅の効率は改善するが、DRAMチップがバンクに編成される方式のために、改善の程度はブロックのサイズに左右される。

【0008】特に、現代のDRAMチップは複数のバンクのメモリからなり、各バンクは複数の列からなり、各列は連続的なメモリアドレスにおける複数のデータエレメントからなっている。かかるDRAMチップは、単一のバンクの単一の列内においてデータを読み書きしている間に、その最大レートの帯域幅を持続することができ

る。同時に、異なったバンクの列は「プリチャージする：pre-charged」か転送の準備をすることができ、従って、読み書きは直前に使用したバンクの直前の列から新たに使用するバンクの新たな列へと中断なく継続して行うことができる。しかし、DRAMチップは同一バンクの2つの異なる列を素早く連続して読み取ったりそこに書き込んだりすることはサポートしていない。すなわち、同一バンクの異なる列から読み出したりそこに書き込んだりしている間に、バンクの1つの列をプリチャージすることは不可能である。DRAMには付加的な制限を課すもの、例えば、使用中のバンクに隣接したバンクのプリチャージを禁ずるものもある。

【0009】同一の、もしくは対立するバンクの異なる列を読み出したりそこに書き込んだりしなければならないような形でデータが編成されている場合には、常に、数サイクルの遅延が課される。実時間ボリュームレンダリングシステムにおいては、この遅延による影響はブロックのサイズに応じて変わる。例えば、EM-Cubeシステムの実施の形態においては、ブロックは $8 \times 8 \times 8$ ボクセル、すなわち合計で512ボクセルである。この場合に、同一バンクの列の間で8サイクルの遅延があるDRAMを使用すると、約97%の効率でDRAMチップからボクセルデータを読み取ることが依然として可能である。しかし、それよりも少ない $2 \times 2 \times 2$ のボクセル、すなわち合計で8ボクセルを有する別の実施の形態においては、DRAM帯域幅の効率は約50%に削減されるであろう。ここで、実時間ボリュームレンダリングシステムに関する試みは、ブロックを十分に大きく保つか、同一のもしくは対立するバンクの異なった列への迅速かつ連続的なアクセスを避けるかの何れかによって、DRAMメモリの効率を最大化するようにデータを編成することである。

【0010】

【発明が解決しようとする課題】上記従来のシステムは、通信効率を最大化するために比較的大きなボクセルデータのブロックを利用しているが、現在単一の集積回路またはチップ上でボリュームレンダリングシステムを実施することが望ましくなっている。しかし、実時間ボリュームレンダリングのパフォーマンスを達成するためには、ボクセルの記憶および分散という根本的なアーキテクチャにおける変更が必要である。

【0011】この発明は上述した点に鑑みてなされたもので、実時間ボリュームレンダリングのパフォーマンスを達成できる実時間ボリュームレンダリングを可能にする方法を提供することを目的とするものである。

【0012】

【課題を解決するための手段】この発明に係る実時間ボリュームレンダリングを可能にする方法は、ボクセルデータをDRAMメモリモジュールに配置して、ボリュームレンダリングシステムにおけるデータ転送中のアイド

ルサイクルを回避し、それによってDRAMメモリモジュールの最大バーストレートでのデータ転送による実時間ボリュームレンダリングを可能にする方法であって、ボクセルデータ集合からのボクセルデータをミニブロックに配置するステップと、それぞれ多数のDRAMメモリバンクを有する多数のDRAMモジュールを提供するステップと、各DRAMモジュール内で、連続的にアクセスされるミニブロックが関連するDRAMモジュール内で異なったバンクに割り当てられるように、ミニブロックをそのメモリバンクに割り当てるステップであり、上記ミニブロックは、特定のDRAMと2層記憶システムにおけるDRAM内の特定のバンクとの両方に割り当てられ、上記バンクの連続的アクセスがあった時点で、関連するDRAMモジュールからのデータ転送の間のアイドルサイクルが回避されるステップとを含むものである。

【0013】また、所定の順序でDRAMメモリのバンクからボクセルデータを読み出すステップと、上記読み出しがコンフリクトにつながる場合を検出するステップと、上記コンフリクトの検出に応じて読み出し順序を反転させるステップとをさらに含むことを特徴とするものである。

【0014】また、上記コンフリクトは、現在読み出されているのと同じメモリバンクからの連続的読み出しを含むことを特徴とするものである。

【0015】また、DRAMメモリから読み出されたボクセルデータを再度順序付けして、上記ボクセルが上記ボリュームデータ集合に配置された順序でのボクセル処理を可能にするステップをさらに含むことを特徴とするものである。

【0016】また、上記ミニブロックのそれぞれは、 $2 \times 2 \times 2$ ボクセルの立体配列での8ボクセルを含むことを特徴とするものである。

【0017】また、上記ミニブロックは、次式に従って上記DRAMモジュールに記憶され、

$$\text{DRAM Module Number} = (X_{mb} + Y_{mb} + Z_{mb}) \bmod M$$
ここで、 $X_{mb} + Y_{mb} + Z_{mb}$ は次式によって求められる座標(X、Y、Z)を有するボクセルを含むミニブロックの位置を定義する。

$$X_{mb} = [X/2], Y_{mb} = [Y/2], Z_{mb} = [Z/2]$$

M個のモジュールの全ては、1つのモジュールにアクセスするのに必要な時間と同じ長さで同時にアクセスでき、ミニブロックの座標を合計し、Mで割って剰余を取ることによって、上記ボリュームデータ集合の何れかの軸と同一線上のM個のブロックの何れかのグループが一斉かつ同時にフェッチされることを保証することを特徴とするものである。

【0018】また、座標(X_{mb}, Y_{mb}, Z_{mb})を有するミニブロックは、次式に従ってバンクに割り当てられる

Bank Number = $[(X_{mb}/M) + (Y_{mb}/M) + Z_{mb}/M] \bmod B$

ここで、Mは独立のメモリチップの数であり、Bは1チップ当たりのバンクの数であり、また、BはDRAMチップ1個当たりのバンクの数である。ことを特徴とするものである。

【0019】

【発明の実施の形態】以下、この発明について説明する。この発明に係るシステムにおいては、DRAMメモリモジュールとの間のデータ転送がほぼ100%のレベルにまで確実に最大化されるように、ボクセルデータのミニブロック編成が採用されている。ミニブロック編成は、各ミニブロックがDRAMメモリモジュールまたはチップに割り当てられ、さらにDRAMメモリモジュール内の特定のバンクに割り当てられるようになっていく。これは、メモリモジュールのみならずそこに含まれたバンク全体に渡ってボクセルデータの2層の割り当てすなわちスキューイングを構成する。メモリアーキテクチャは、削減されなければ、DRAMモジュールの定格帯域幅の50%以下にまで大幅にデータレートを減少させてしまう、アイドルサイクルの数を最少値にまで削減する。

【0020】ボクセルデータはミニブロックに記憶されているので、DRAMモジュールの最大データレートバーストで読み出すことができる。ミニブロックはDRAMのバンク全体に渡って分散されるかスキューされるので、メモリバンクをプリチャージするのに必要なアイドルサイクルを生じさせることなく、連続したミニブロックを読み出すことが可能になる。周知のように各アイドルサイクルは、それがなければデータを読むことができる（実際には読めない）損失時間の単位を表す。このアイドルサイクルは、メモリから読み出すデータの効率を削減し、実時間ボリュームレンダリングシステムの費用を増加させる、もしくは全く不可能にしてしまう。従って、このアイドルサイクルを回避することが重要である。

【0021】各DRAMチップのバンク全体に渡るミニブロックのスキューイングをもってしても、1つの列の末端で1つのDRAMの1つのバンクからミニブロックをフェッチした直後に、次の列の始点で同一DRAMモジュールの同一バンクからミニブロックを連続してフェッチしてしまうといった事態がしばしば発生する。この問題を回避するために、この発明は、読み出される次のミニブロックがDRAMメモリの同一または対立するバンクからのフェッチにつながる場合には、ミニブロックの列の読み出し方向を反転するシステムを利用している。

【0022】ボクセルデータは割り当ての順序でDRAMメモリモジュール全体に渡って分散されるのであって、特定の観察方向に関する通常の処理ではないので、

DRAMモジュールの出力を再配置して所与の観察方向に関する通常の処理順序を再確立するために、特化されたデスキューイング回路が提供される。これは、一つの実施の形態においては多くの内部バッファおよびアドレスリングシステムの制御下でのシフティングまたはマルチプレクシング回路によって達成される。

【0023】より詳細には、この発明においては、実時間ボリュームレンダリングに必要とされるパイプライン化された処理エレメントは、単一の半導体チップに設けられた単一の集積回路に含まれている。他の処理チップとの通信はなく、従って、かかる通信の帯域幅を管理する必要もないので、大規模なブロックの必要性は抹消される。従って、この発明は、DRAMチップのバーストモード転送を利用することができる最小のブロックを使っている。より好適な実施の形態においては、ブロックは $2 \times 2 \times 2$ であり、従ってミニブロックと呼ばれている。ミニブロックは、前述のEM-Cubeボリュームレンダリングシステムにおけるのと全く同じ方法で、メモリのミニブロックの位置を示す同一の数学的公式をもって、ボリュームデータ集合全体に渡ってスキューされる。

【0024】DRAMの同一バンクからの連続的なフェッチの結果生ずるDRAM帯域幅の効率低下を避けるために、この発明は、ミニブロックがDRAMチップ全体に渡ってスキューされ、また各DRAMチップ内のバンク全体にも渡ってスキューされる、2層のスキューイング手法を利用している。すなわち、隣接するミニブロックが別個のDRAMチップに記憶され、各DRAMチップ内においては、ボリュームデータ集合の3本の軸に平行な何れかの方向で、至近隣接するバンクから離れたバンクにミニブロックが記憶されている。レンダリングの間には、ミニブロックはグループでフェッチされ、グループ内のミニブロックの数は、システムにおいて同時にアクセス可能なDRAMチップの数と同じである。ミニブロックのグループは、例えば好適な実施の形態において、ボリュームデータ集合が画像平面から観察される時に左から右、上から下、正面から背面へとステップすることにより、秩序だった方法でフェッチされる。2層のスキューイングによって、何れかのDRAMチップにおける連続的なミニブロックは異なったバンクからフェッチされ、従って、DRAMの帯域幅の効率が最大化されることが確実になる。

【0025】しかし、ミニブロックの各配置、スキューイングの方法、およびボリュームデータ集合を介した処理の順序に関しては、ミニブロックの列の最後のミニブロックが、ミニブロックの次の列の最初のミニブロックと同じバンクに記憶されるような、少なくとも1つの方向付けまたは観察方向がある。特別の配慮がなければ、このような1つの事態が処理の遅延、DRAM帯域幅の浪費効率、またはパイプライン化された処理エレメント

における付加的な複雑性を発生させることになる。したがって、この発明は、これらの特定の列を反対の順序で処理することにより、このような事態を補正するものである。すなわち、例えば左から右へとミニブロックの列がフェッチされ、次の列が前の列と同一バンクにあるミニブロックで始まる場合には、次の列は右から左にフェッチされる。この手段によって、ミニブロックへの全ての連続的なフェッチは、DRAMチップのコンフリクトしない、異なるバンクから行われることになる。

【0026】ミニブロックがその対応するDRAMチップからフェッチされるときに、デスキューイング回路、すなわち観察方向に従ってボクセルを再配置し、各ボクセルをそれを処理する処理パイプラインで整列させる回路を通過させられる。幾つかの実施の形態においては、各ボクセルは正確に1つの処理パイプラインと関連づけられており、隣接するボクセル値は、隣接する処理パイプラインからまたは処理チップのFIFOを介して再循環されるボクセルデータからの通信によって得られる。しかし他の実施の形態においては、ボクセルキャッシュとよばれる、処理チップ上の記憶バッファが利用される。これらの実施の形態においては、パイプラインは互いには全く通信せず、何らかの特定の処理に必要なボクセルをボクセルキャッシュからフェッチする。何れのアプローチにおいても、デスキューイング回路およびボクセルキャッシュが、各処理パイプラインが正しいデータを正しい順序で検索することを可能にしている。

【0027】要約すると、2層のスキューイングアーキテクチャが、ボクセルデータが1組のDRAMメモリモジュールに割り当てられたミニブロックに記憶されている、ボリュームレンダリングシステムのメモリサブシステムに課されており、それによって、DRAMメモリの最大バーストレートでのデータ転送を可能にし、実時間ボリュームレンダリングを可能にしている。各DRAMモジュール内においては、連続的にアクセスされるミニブロックが異なったバンクに割り当てられるように、ミニブロックが複数のメモリバンクに割り当てられており、それによって、データ転送中のアイドルサイクルを回避してDRAM転送効率をほぼ100%に増加させる。一つの実施の形態においては、DRAMメモリのバンクからのボクセルデータの読み出しは、バンクのコンフリクトがなければ左から右に進行し、バンクのコンフリクトがあれば読み出し順序は反転する。特化されたデスキューイング回路は、ボクセルがメモリに記憶されている順序ではなく、ボリュームデータ集合に配置された順序でボクセルを処理できるように、DRAMメモリから読み出されたボクセルデータを再配置するために提供する。

【0028】より詳細には、低コストな実時間ボリュームレンダリングシステムの記憶サブシステムは、バーストモードのダイナミックランダムアクセスメモリ(DR

AM) 集積回路または「チップ」のピークレートに近似の持続するデータレートでボリュームエレメントまたは「ボクセル」を配布する。このデータレートは観察方向、すなわちボリュームデータ集合を通過する光線の方法からは独立しており、システムにおけるDRAMチップの数に比例して一次的に増減する。三次元のボクセルデータは $2 \times 2 \times 2$ のサイズのミニブロックに編成される。観察方向からの独立性は、隣接するミニブロックが隣接するDRAMチップに記憶されるように、DRAMモジュール全体に渡ってミニブロックをスキューすることにより達成される。これは、あらゆるクロックサイクルにおいて全てのチップから並行してデータを読み取ることができること、および、データは再度読み取る必要がないことを保証し、それによってメモリ帯域幅を最大化する。各DRAMチップ内においては、ミニブロックはそのチップのメモリバンク全体に渡ってスキューされる。これは、同一バンクの異なるページへの連続的なフェッチにつきもののプリチャージおよび再起動時間を削減する。一次的に増減するデータレートのスケラビリティは、システムにおけるDRAMモジュールの数を増加させることにより達成できる。ボリュームレンダリングシステムのメモリサブシステムは、パイプライン毎にDRAMチップサイクルにつき1ボクセルのレートでレンダリングパイプラインにボクセルを提供するために、ミニブロックをデスキューする。

【0029】パイプライン化されたボクセルレンダリング

図1を参照しさらに背景を説明すると、三次元ボリュームデータ集合10が図示されている。図1は矩形の立体の形状で配置されたボクセル位置12を示す。より詳細には、ボクセル位置は三次元で立体を満たしており、均一に間隔が隔てている。各ボクセルの位置は、X、YおよびZという3本の軸11によって定められた座標系で表示することができる。各ボクセル位置には、例えばオブジェクトまたはシステムの内部全体および近傍の空間の別個の点における密度、材料の種類、温度、速度、不透明度や他の属性などの、検討中のオブジェクト、システムまたは事象の幾つかの特徴を表す1つ以上のデータ値が関連づけられている。コンピュータ内部では、配列インデックス位置(X、Y、Z)を三次元空間の座標(X、Y、Z)におけるボリュームデータ値に対応させ、三次元配列の値としてボリュームデータ集合をコンピュータに表示する方法が便利である。

【0030】図2は、人間の頭部の断層撮影スキャンによるスライスからなるボリュームデータ集合10の例を示す。二次元画像平面16は、人間の頭部のボリュームレンダリングされた投影が表示された表面を示している。光線キャスティングとして公知の技術においては、光線18がボリュームデータ集合10を通過して画像平面16のピクセル位置22から投影され、各光線はポリュ

ームを通過するときに、ボクセル位置のデータから色と不透明度を蓄積する。このようにして、色、透明度、輝度並びにピクセルのその他の属性が、光線に沿ったサンプルポイント20におけるデータの蓄積としてポリウムデータ集合から抽出される。この例においては、骨組織と関連するボクセル値には不透明色が割り当てられ、頭部の他の全ての組織に関連するボクセル値には透明色が割り当てられる。従って、光線に沿ったデータの蓄積および、対応するピクセルへのこのデータの帰属は、実際の頭蓋は皮膚や頭部の他の組織によって視界からは隠れていても、観察者には三次元の頭蓋の画像に見える画像19を視察面16に形成することになる。

【0031】光線キャスティング(ray-casting)の方法をより完全に理解するために、図3は図2の三次元ポリウムデータ集合10の二次元断面を示している。第一および第二の次元はこのページの平面に図示された次元に対応する。ポリウムデータ集合10の第三の次元は、データ集合の断面のみを図において見ることでできるように、印刷されたページに垂直である。ボクセル位置は図の点12によって示されている。各位置に関連したボクセルは、三次元空間における矩形グリッドの固定点での三次元オブジェクト14のある特徴を表すデータ値である。図3には、適当な特徴を備えたピクセル22を提供するという意味で、オブジェクト14の画像が投影される二次元画像平面16の一次元の図を示す。この図においては、画像平面16の第二の次元も印刷されたページに垂直である。

【0032】光線キャスティングの技術において、光線18は画像平面16のピクセル22からポリウムデータ集合10を通過して伸びている。各光線は、その光線に沿ってサンプルポイント20において色、輝度および透明度または不透明度を蓄積する。この光の蓄積が、対応するピクセル22の明るさおよび色を決定する。従って、ポリウムを通過してピクセルから外方に進むと図示されている一方で、蓄積されたデータは光線を戻って送信され、対応するピクセルにパラメータ、とりわけピクセルの色、輝度および不透明度または透明度を与えられるように対応するピクセルに提供されることが考えられることができる。

【0033】図3はポリウムデータ集合10の第三の次元と画像平面16の第二の次元が、共に印刷ページに垂直であり、従って互いに平行であると示唆しているが、一般的にはそうではないということである。画像平面はポリウムデータ集合に対していかなる方位も有することができる。従って光線18は三次元すべてにおいてどんな角度でもポリウムデータ集合10を通過する。

【0034】また、サンプルポイント20はボクセル12の座標と必ずしも正確に交差するとは限らないことがわかる。従って、各サンプルポイントの値は、隣接するボクセルの値から合成しなければならない。すなわち、

各サンプルポイント20における光の輝度、色および透明度または不透明度を、隣接するボクセル12の値の数学的関数によって計算するか、もしくは補間する必要がある。ボクセルデータ値のサンプルポイントにおける値への再サンプリングは、サンプリング理論として知られる数学部門の応用である。そして、各光線18のサンプルポイント20は、もうひとつ数学的関数によって蓄積され、その光線に対応するピクセル22の輝度および色が求められる。その結果得られるピクセル22の集合は、オブジェクト14の視覚画像を画像平面16に形成する。

【0035】図4は個々の光線の処理を示している。光線18はボクセル位置12の付近またはその位置を通過して、三次元ポリウムデータ集合10をある角度で通過し、各光線に沿ったサンプルポイント20でデータを蓄積する。21に示すように、各サンプルポイントにおける値は、補間(内挿)ユニット103(図5参照)によって合成され、その勾配は23に示すように勾配推測ユニット111(図5参照)によって計算される。サンプルポイント20からのサンプルポイント値と各サンプルポイントに関する勾配25は次に、パイプラインにおいて処理されて各サンプルに色、輝度または強さ、および透明度または不透明度を割り当てる。27に示すように、これはパイプライン処理を経て、赤、緑および青の色相ならびに輝度および透明度または不透明度が計算される。最後に、全ての光線のすべてに沿ったサンプルのすべてに割り当てられる色、輝度レベルおよび透明度が、29に示すように合成ユニット124に適用される。合成ユニット124は、これらのサンプル値を数学的に結合してピクセルと成し、ピクセルは結果として生ずる画像32を画像平面16上に描く。

【0036】サンプルポイント20の色、輝度または強さおよび透明度の計算は、2つに分けて行われる。第一に、三線補間の数学的関数を利用して、サンプルポイント20を隣接して囲むキュービック配置内の8つのボクセルの値の加重平均を取る。その結果得られる平均はを使って、ある伝達関数によって色および不透明度または透明度をサンプルポイントに割り当てる。第二に、隣接のサンプルポイント間の差を取ることにによって各サンプルポイント20におけるサンプル値の数学的勾配を概算する。理解されるように、これら2つの計算は数学的に均等な結果を得るために、順番にもしくは互いに並行して実行することが可能である。勾配は次に、サンプルポイントの明るさを判断するための照明計算に使用される。照明計算はコンピュータグラフィックスの分野においては周知であり、例えば、Addison-Wesley of Reading, Massachusettsによって1990年に発行された、J. Foley, A. vanDam, S. FeinerおよびJ. Hughes著の教科書「Computer Graphics: Principles and Practice」第2版で説明されている。

【0037】図5は、図4に示した計算を行うのに適したパイプライン化されたプロセッサのブロック図を示している。パイプライン化されたプロセッサは複数のパイプライン段階からなり、各段階は1つのデータエレメントを保持しており、従って、複数のデータエレメントが一度に処理される。各データエレメントはその処理においては進捗度合いが異なっており、全てのデータエレメントがパイプラインの段階から段階へとロック（固定）ステップで移動する。パイプラインの第一の段階では、一連のボクセルデータ値は1サイクル当たり1ボクセルのレートでボクセルメモリ100からパイプラインに流れ込み、アドレスジェネレータ102の制御下で機能する。補間（内挿）ユニット104は、三次元空間のX、YおよびZ座標にあるボクセル値を受け取る。ここでX、YおよびZはそれぞれ整数である。補間（内挿）ユニット104は、パイプライン化された一組の段階であり、ボリュームを通して放たれる光線に沿った位置に対応するボクセル間のデータ値をサンプルポイントで合成する。各サイクルの間に、1つのボクセルが補間（内挿）ユニットに入ると、1つの補間（内挿）されたサンプル値が現れる。ボクセルがパイプラインに入ってから補間（内挿）されたサンプル値が現れるまでの待ち時間は、パイプライン段階の数と各段階での内部遅延に左右される。

【0038】パイプラインの補間（内挿）段階は、一組の補間（内挿）段階104と、段階を経てデータを再循環するための3個のFIFOエレメント106、108、110からなる。一つの実施の形態においては、これらは全て線形補間（内挿）であるが、立方（cubic）やラグランジアン（LaGrangian）などの他の補間（内挿）関数を採用してもよい。図示した実施の形態においては、補間（内挿）は各次元において個別の段階として行われ、空間では隣接しているがパイプラインに入る時点では全く離れているボクセル間で補間を行えるよう、データを再循環するための個々のFIFOエレメントが含まれている。各FIFOの遅延は、特定の次元における1つのボクセルの読み取りと隣接するボクセルの読み取りとの間に経過した時間の長さに正確に一致するよう選択され、これら2つのボクセルが補間関数において結合できるようになっている。理解されるのは、各ボクセルがその次元に関連したFIFOを通してあらかじめ再循環された最も近接するボクセルと結合下状態で、1サイクル当たり1ボクセルのレートでボクセルを補間段階に通過させることができる。

【0039】各次元に1つずつの3つの連続的な補間段階は連結されており、ボクセルは入力と出力の両方において、1サイクル当たり1ボクセルのレートでこれら3つの段階を通過することができる。補間段階のスループットは、補間（内挿）ユニット内の段階の数に関係なく、また、補間（内挿）ユニット内のデータ待ち時間お

よびユニット内の再循環段階の待ち時間に関係なく、1サイクル当たり1ボクセルである。従って、補間（内挿）ユニットは1サイクル当たり1ボクセルのレートで、XYZ空間の整数位置にあるボクセル値を、非整数位置にあるサンプル値に変換する。より詳細には、補間（内挿）ユニットは、ボクセル位置の値を光線に沿って配置されたサンプル位置の値に変換する。

【0040】補間（内挿）ユニット104に続くのは勾配推測ユニット112であり、これも複数のパイプライン化された段階と再循環FIFOからなっている。勾配ユニット112の機能は、3つの各次元におけるサンプル値の変化レートを推測することである。勾配推測ユニットは補間（内挿）ユニット104と同様の方法で機能し、3つの次元のそれぞれにおけるサンプル値の変化レートを計算する。勾配は照明に関する通常のベクトルを判断するのに使用され、その大きさは、勾配の程度が高い場合に表面の存在を測るものさしとして使用することがある点に注意して戴きたい。本実施の形態においては、中央の差を取るにより得られるが、当該技術分野で公知な他の関数を採用してもよい。

【0041】勾配推測ユニットはパイプライン化されているので、1サイクル当たり1つの補間されたサンプルを受信し、1サイクル当たり1つの勾配を出力する。補間（内挿）ユニット同様、各勾配は、個々の再循環FIFO114、116、118を含む勾配推測ユニット112における待ち時間の長さに等しいサイクルの数によって、その対応するサンプルから遅延する。各再循環FIFOの遅延は、その次元において勾配を演算するのに必要な1つの補間されたサンプルと近接する付近の補間されたサンプルと、読み取りに必要な時間の長さによって決定される。

【0042】補間されたサンプルとそれに対応する勾配は、1サイクル当たり1つの補間されたサンプルにつき1つの勾配のレートで、それぞれ分類ユニット120および照明ユニット122に同時に適用される。分類ユニット120は、補間されたサンプル値をグラフィックシステムにおける色、すなわち赤、緑、青と、RGBA値としても知られるアルファ値に変換する役割を果たす。赤、緑および青の値は通常、ゼロから1までの間の分数であり、個々の補間されたサンプル値に割り当てられる色部品（カラーコンポーネント）の輝度を表す。アルファ値も通常、ゼロから1までの間の分数であり、個々の補間されたサンプル値に割り当てられる不透明度を表す。

【0043】勾配は照明ユニット122に適用されて、強調と影を付加することにより新たに割り当てられたRGBA値を調整して、より現実的な画像を提供する。照明を行う方法と機能は当該技術分野においては公知である。照明ユニットおよび分類ユニットは1サイクル当たり1つの補間されたサンプル値と1つの勾配を受け入

れ、1サイクル当たり1つの照明された色 (illuminated color) と不透明度の値を出力する。

【0044】当該の実施の形態においては、補間 (内挿) ユニット104は勾配推測ユニット112より先行し、勾配推測ユニット112は分類ユニット120より先行する、他の実施の形態においてはこれら3つのユニットは異なる順序で配置してもよいことが理解される。より詳細には、ボリュームレンダリングの幾つかの適用において、分類ユニットは補間 (内挿) ユニットに先行することが望ましい。この場合に、ボクセル位置のデータ値は同じ位置でRGBA値に変換され、次にこれらのRGBA値は補間されて光線に沿ったサンプルポイントでRGBA値を得る。

【0045】合成ユニット124は光線に沿って全てのサンプルポイントの照明された色 (illuminated color) および不透明度の値を結合し、コンピュータ端末または二次元画像表面に表示するために、その光線に対応する最終ピクセル値を形成する。RGBA値は1サイクル当たり1RGBA値のレートで合成ユニット124に入り、同じ光線に沿った以前のサンプルポイントにおけるRGBA値とともに蓄積される。蓄積が完了すると、最終の蓄積値がピクセルとしてディスプレイに出力されるか画像データとして記憶されるかする。合成ユニット124は1サイクル当たり1RGBAのサンプルを受けとり、光線の終端に達するまで合成機能に従って光線ごとこれらの光線を蓄積し、光線の終端に達したところで光線当たり1ピクセルを出力して最終画像を形成する。適用の形態によっては、合成ユニットには当該技術分野で公知の多くの異なる機能を採用することができる。

【0046】照明ユニット122と合成ユニット124との間には、多様な調整ユニット126を設けて照明されたRGBA値の調整を可能にし、それによって最終的に見る画像を調整することができる。かかる調整ユニットの1つは、サンプル値を切り落としてデータの限定されたサブセットを見ることを可能にするのに使用される。他の調整ユニットは、任意の角度および厚さでボリュームデータのスライス部分を表示す機能を提供する。3番目の調整ユニットは、三次元カーソルを提供して、ユーザーまたはオペレータがデータ内のXYZ空間で位置を特定することを可能にする。上記の機能のひとつひとつが、1つのRGBA値を1サイクル当たりの入力として受入れ、1サイクル当たり1つの調整されたRGBA値を出力として放出する複数のパイプライン化された段階として実施される。本明細書中に記載のパイプライン化されたアーキテクチャ内では、同様に実施することができる他の調整機能も提供され得る。パイプライン化された調整機能を付加することは、処理パイプラインのスループット (レート) を減少させるようなことはいっ

さいなく、データがパイプラインを通過する際のその待ち時間に影響する。

【0047】例えば、 $256 \times 256 \times 256$ ボクセルを有するボリュームデータ集合についていえば、1秒当たり30フレームの実時間ボリュームレンダリングレートを達成するためには、ボクセルデータは、1秒当たり $256^3 \times 30$ フレームまたは1秒当たり約5億ボクセルでパイプラインに入らなければならない。分かることは、特定のボクセルのいずれに関係する計算も多くの段階を含み、従って定められた待ち時間を有するが、各ボクセルが異なる進捗段階にあり、パイプラインの異なる段階を占めている状態で、複数の異なるボクセルに関連した計算を直ちに行うことが可能である。これによって、計算の複雑さにも拘わらず高い処理レートを維持することが可能になる。

【0048】さらに、上記のパイプライン化されたプロセッサを複数の平行なパイプラインとして複製して、隣接するボクセルを並行に処理することによりさらに高いスループットレートを達成できることが理解されるであろう。各パイプラインのサイクル時間は、通常のボリュームデータ集合内のボクセル数に求められるフレームレートを掛けパイプライン数で割れば求められる。好適な実施の形態においては、サイクル時間が7.5ナノ秒であり、4つのパイプラインが並行に採用され、それによって1秒当たり5億ボクセル以上もの処理レートを達成する。

【0049】ミニブロック

図6および図7を参照すると、この発明は上記の米国特許出願第08/905,238号に記載のEM-Cubeのブロックングの方法を利用しており、この方法では、一つの実施の形態において、各ブロックは「ミニブロック」とも呼ばれる $2 \times 2 \times 2$ のサイズの立体的配列からなる。図6は軸206の座標系に従って三次元空間に配置された8個の隣り合うボクセル202の配列200を示す。8個のボクセル202のデータ値は、メモリ208の8エレメント配列に記憶される。各ボクセルは、座標(X、Y、Z)によって表された三次元空間における位置を占め、X、YおよびZは全て整数である。ミニブロックのメモリ配列内のボクセルデータ値のインデックスは、X、YおよびZ座標それぞれの順序が下位のビットから判断される。図6に示すように、これら3つの下位の順序のビットは連結されて、ゼロから7までの値となる3ビット二進数204を形成し、これは次にそのボクセルに対応する配列エレメントを識別するのに利用される。換言すれば、座標(X、Y、Z)でのボクセルのデータ値のミニブロック内における配列インデックスは、

$$(X \bmod 2) + 2 \times (Y \bmod 2) + 4 \times (Z \bmod 2) \quad (1)$$

によって求められる。

【0050】各ボクセルまたはサンプルの位置が座標

(X、Y、Z)によって三次元空間で表すことができるように、ミニブロックの位置はミニブロック座標

(X_{mb} 、 Y_{mb} 、 Z_{mb})で表すことができる。これらの座標においては、 X_{mb} はX軸に沿ったミニブロックの位置を表し、全ミニブロックのユニットで数える。同様に、

$$X_{mb} = [X/2], Y_{mb} = [Y/2], Z_{mb} = [Z/2] \quad (2)$$

によって求められる。

【0051】図7を参照すると、ミニブロックスキューイングの最初の層が図示されている。これは次式に従つ

$$DRAM\ Number = (X_{mb} + Y_{mb} + Z_{mb}) \bmod 4 \quad (3)$$

図においては、ミニブロック200の三次元配列の部分図が示されており、各ミニブロックは番号を付けた小さな立体によって図示されている。番号はそのミニブロックの特定のDRAMモジュールチップへの割り当てを表す。図示した実施の形態においては、0、1、2および3と番号を付された4個の異なるDRAMチップがある。軸と整列した4つのミニブロックの各グループは、4つの番号のそれぞれが付いた1つのミニブロックを含むことが、図から理解される。これは式(3)から確認することができる。すなわち、座標(X_{mb} 、 Y_{mb} 、 Z_{mb})の何れかのミニブロックで始めて、X軸方向にミニブロックを通して連続すると、式(3)のDRAM Numberは数字0、1、2および3を経て継続的にサイクルする。同様に、Y軸またはZ軸に平行なミニブロック

$$Module\ Number = (X_{mb} + Y_{mb} + Z_{mb}) \bmod M \quad (4)$$

すなわち、M個全てがモジュールにアクセスするのに必要な時間と同じ時間で同時にアクセスすることができるように、図示した実施の形態のメモリサブシステムが、M個の別モジュールからなっているのであれば、ミニブロックのメモリモジュールへの割り当ては、ミニブロックの座標を合計してMで割り、剰余を取るによって求められる。これは、何れかの軸と一線に並んだM個のブロックの何れかのグループを、同時にフェッチすることができることを保証する。ボリュームデータ集合の何れかの軸に沿ってM個のミニブロックのグループを同時にフェッチする必要があるのは、ボリュームデータ集合の横断の順序が観察方向に依存するためである。

【0053】図示した実施の形態においては、ミニブロックはボリュームデータ集合の軸と一線上に並んだ線形グループにおいてアクセスされるが、ボリュームデータ集合の横断の順序とは関係なく、矩形グループ、立体グループまたは他の大きさおよび形状のグループにおいてもフェッチできるように、他の実施の形態では異なる公式によってミニブロックをスキューイングすることが理解されるであろう。

【0054】現代のDRAMチップでは、そのDRAMのタイプに応じたクロックレートで、かつ適度な大きさのバーストで、DRAMチップからデータをフェッチしたりDRAMチップにデータを書き込んだりすることが可能である。いわゆるシンクロナスDRAMまたは「S

Y_{mb} と Z_{mb} はY軸およびZ軸にそれぞれ沿ったミニブロックの位置を表し、全ミニブロックで数える。このようなミニブロック座標の表示を使うと、座標(X、Y、Z)を有するボクセルを含むミニブロックの位置は、

たEM-Cubeからのブロックのスキューイング方法の応用である。

を経て連続することにより、式(3)もDRAM Number 0、1、2および3を経て継続的にサイクルする。従って、3本の軸の何れかに平行な何れかの方向209、211または213でミニブロックの三次元配列を横断する場合には、4つの隣接するミニブロックのグループは常に、DRAMチップの4つの独立したメモリから並行にフェッチできることが理解される。ミニブロックのメモリモジュール内のメモリロケーションへの割り当てを以下に説明する。

【0052】より一般的には、システムがM個の独立したメモリモジュールを含む場合は、座標(X_{mb} 、 Y_{mb} 、 Z_{mb})を有するミニブロックは次のようにメモリモジュールに割り当てられる。

DRAM」と呼ばれるチップには、133MHz、147MHzおよび166MHzの各種類が含まれており、通常のクロックレートは、1サイクル当たり7.5ナノ秒、7ナノ秒および6ナノ秒にそれぞれ対応する。クロックレートを維持するのに必要な通常のバーストサイズは、一個あたり16ビットの5個から8個のメモリエレメントである。開発中の別の種類のDRAMは800MHzでのクロックレートと、各16ビット、通常16個のデータエレメントのバーストサイズを有する。これら現代のDRAMチップにおいては、DRAMチップ内の独立したメモリバンクからであれば、連続的なバーストはアイドルサイクルの介入なしに適合できる。すなわち、連続してアドレスされたデータエレメントのグループは、DRAMチップの異なる、もしくは対立(コンフリクト)しないメモリバンクに記憶され、次にこれらはDRAMの最大定格速度で、アイドルサイクルの介入も一切なしに、素早く連続して読み取り、書き込みを行える。

【0055】図8を参照すると、ミニブロックはDRAMのバンクに対応するグループにさらに配置されている。これはボクセルスキューイングの第二の層を構成する。4x4x4のミニブロックの各グループは大きな数字を付されている。各数字は、そのグループの各ミニブロックの、その割り当てられたDRAMチップにおける同じ数字を有するバンクへの割り当てを示している。例

例えば、図のミニブロック212のグループは数字0が付されている。これは、グループ212内の各ミニブロックは、メモリチップごとにあるバンク0に記憶されることを意味する。同様に、グループ214のミニブロックの全てが、メモリチップごとにあるバンク1に記憶され、グループ216の全てがメモリチップごとにあるバ

$$\text{Bank Number} = \{ [X_{mb}/4] + [Y_{mb}/4] + [Z_{mb}/4] \} \bmod 4 \quad (5)$$

図示した実施の形態におけるDRAMチップ毎のバンクの数と、DRAMチップの数とが同数であるという事実は単なる偶然である。

【0057】図から、所与の直交する方向において、一組のパイプライン化された処理エレメントがボリュームデータ集合を横断する場合には、4つのミニブロックを何れかの軸に平行なグループで一度にフェッチすると、隣接するグループ0およびグループ1などのグループは常に異なるバンクにあることが理解されるであろう。これは、4つのミニブロックのグループが、DRAMチップの「バーストモード: burst mode」アクセスを利用して、DRAMチップの部分にアイドルサイクルを介入させずに、またどの軸に沿って横断しても素早く連続してフェッチすることを意味する。また、これはDRAM帯域幅の効率を最大化する。

$$\text{Bank Number} = \{ [X_{mb}/M] + [Y_{mb}/M] + [Z_{mb}/M] \} \bmod B \quad (6)$$

しかし、例えば3つの次元における距離が互いに相対的な基本となるように、異なる距離により各次元でスキューするというように、他の実施の形態では他の規則にしたがって、バンク全体に渡ってミニブロックをスキューしてもよいことが理解されるであろう。

【0059】図示した実施の形態においては、ミニブロックは次のようにそのDRAM内の特定のメモリアドレスに割り当てられる。ボリュームデータ集合が、X軸に沿った幅でSXミニブロック、Y軸に沿った高さでSY

$$\text{Mini Block Number} = X_{mb} + Y_{mb} \times SX + Z_{mb} \times SX \times SY \quad (7)$$

このDRAM NumberおよびBank Numberへの割り当ては、式(3)および(5)によってそれぞれ求める。次

$$\text{Mini Block Index In Bank} = [\text{Mini Block Number} / 16] \quad (8)$$

すなわち、4つのDRAMモジュールがあり、それぞれに4つのバンクがあり、合計で16個のバンクがある。従って、バンク内のミニブロックの位置は、そのMini-block Numberをバンクの総数で割ったものである。最後に、図示した実施の形態においては、バンクの各列は3

$$\text{row} = [\text{Mini Block Index In Bank} / 32] \quad (9)$$

$$\text{offset In Row} = \text{Mini Block Index In Bank} \bmod 32 \quad (10)$$

この発明の趣旨の範囲内の他の公式によって、他の実施の形態でミニブロックのメモリバンク内のアドレスへの割り当てを行ってもよいことが理解されるであろう。

【0060】横断順序: Traversal Order

この発明においては、観察方向に平行な光線が、メモリのボリュームデータ集合の方向には関係なく、ボリュームデータ集合を前から後ろ、左から右、上から下の方向

ンク2に記憶される。

【0056】図示した実施の形態においては、各DRAMモジュールは4つのバンク、すなわち0、1、2および3を付されたバンクを有している。座標(X_{mb}, Y_{mb}, Z_{mb})を有するミニブロックは、以下の公式に従ってバンクに割り当てられる。

【0058】より一般的には、ミニブロックのメモリバンクへの割り当ては、ミニブロックのメモリチップへの割り当てと類似の方法でスキューできる。換言すると、ボリュームデータ集合がどの方向に横断されていても同時アクセスが可能のように、ミニブロックはM個のチップ全体に渡ってスキューすることができる。同様に、各チップ内のミニブロックはB個のメモリバンク全体に渡ってスキューすることができ、従って、バンク内の連続したミニブロックへのアクセスは、アイドルサイクルの介入によって遅延することはない。これは、チップとバンク全体に渡るミニブロックの2層のスキューイングを形成する。図示した実施の形態においては、ミニブロックのメモリバンクへの割り当ては次の公式によって求める。

ミニブロック、Z軸に沿った深さでSZミニブロックであると仮定する。ここで、これは合計SX×SY×SZ個のミニブロックからなる。さらに、SX、SYおよびSZのそれぞれが、DRAMチップの数にチップ毎のバンクの数を掛けた倍数であると仮定する。すなわち、図示した実施の形態においては、SX、SYおよびSZのそれぞれは16の倍数である。次に、座標(X_{mb}, Y_{mb}, Z_{mb})を有するミニブロックのMini Block Numberを次のように定義する。

に、そのバンク内のミニブロックのインデックス、Mini Block Index In Bankは次の式によって定義される。

2個のミニブロック、すなわちそれぞれ16ビットの32×8個のボクセル、つまり合計4096ビットを保持することができる。従って、列内の列番号とオフセットは次の式によって求められる。

に常に通過するように、ボリュームデータ集合はレンダリングの前に再方向付けされる。これは、ボリュームデータ集合のボクセルの横断を定義する。詳細には、後に処理されるボクセルから得られるサンプルを、先に処理されたボクセルのサンプルの後に合成できるように、横断は光線の方向に従わなければならない。従って、ボクセルの横断は、観察方向によってはボリュームデータ集

合の任意の角で始まってよく、同様に観察方向によっては何れかの軸に沿ったM個のミニブロックのグループで進行してもよい。

【0061】上記のバンクスキューイングの方法は、観察方向に拘わらず、何れの列内においても同一チップの同一バンクから連続的に2つのミニブロックをフェッチする状況を回避しているが、一般的に1つまたは2つの例外となる事態がある。これらは、ミニブロックの1つの列の終端からミニブロックの次の列の始点に横断することから生ずる。例を図8に示す。X軸に平行またはZ軸に平行なグループでミニブロックをフェッチする場合には例外はないが、Y軸に平行なグループについては例外がある。これは、原点がポリウムデータ集合の右上角になければならず、横断がまず正のY方向に進行し、次に負のX方向に、最後に正のZ方向に進行しなければならないような観察方向の場合に発生する。すなわち、ポリウムデータ集合を横断する主たる方向は図8の破線300に平行である。横断はグループ1、2、3および0のミニブロックを通して下方に進み、そのパスに沿って個々のボクセルを読み取る。この横断順序におけるミニブロックの次の列を破線302で示すが、この次の列も図の上から下に進行する。破線300に沿った横断はバンク0で終わり、破線302に沿った横断は同じバンクで始まることが図から分かる。これはもちろん、禁止された一連のアクセスであり、パイプラインに挿入される余分なアイドルサイクルか、DRAMチップのアクセス使用違反につながる。しかし、横断は破線302から破線304に進むので、例外はないことに注意されたい。

【0062】例外的な事態に対する解決策は、横断の間にそれらを見つけて、ポリウムデータ集合の違反している列を、その前の列の反対方向に横断させることである。これを行うメカニズムを以下で説明する。

【0063】デスキューイング：De-skewing
各ミニブロックは、そのメモリチップおよびバンクから1組の連続的なメモリアドレスとして読み取られる。従って、ミニブロックからボクセル値を読み取る順序は、必ずしもボクセルが処理される順序に対応しないことが理解されよう。この状況を考慮するため、デスキューイングの方法を以下に紹介する。

【0064】ここで図9を参照すると、M個のミニブロック群からなるボクセルデータ値を再配置して、それらを正しい横断順序でポリウムレンダリングシステムの並行処理パイプラインに提供するための、デスキューイング回路を示してある。図9の上部では、M個の独立したDRAMチップ230が図5のボクセルメモリ100から構成されている。ミニブロックはアドレスジェネレータ102の制御下でこれらM個のチップから同時に読み取られる。アドレスジェネレータ102はポリウムデータ集合の横断の順序で、ミニブロックのメモリアド

レス234を生成する。DRAMチップ230からのメモリ入力は、選択信号238を介してやはりアドレスジェネレータ102の制御下で機能する1組の選択ユニット236に連結されている。M個のミニブロックはそれに対応するメモリモジュール230から読み取られるので、選択ユニット236が、それらがどのメモリモジュールから来たかに拘わらず、その左から右への整列がポリウムデータ集合におけるミニブロックの物理的位置に対応するように、それらを効率的に再配置または並べ換える。すなわち、各選択ユニット236は、DRAMチップの多くても1つからのその入力を選択し、各DRAMチップ230は多くても1つの選択ユニットによって選択される。

【0065】選択ユニット236の出力は次に、ミニブロックデスキューイングユニット240に連結される。信号線242を介してアドレスジェネレータ102の制御下で機能しているので、各ミニブロックデスキューイングユニットは、データ値が横断の順序に関連した各ボクセルの物理的位置に対応する順序、例えばその自然な順序で提供されるように、そのミニブロック内でデータ値を再配置する。ボクセル値の合計P個のストリームは、ミニブロックデスキューイングユニットから出力され、図5に示された種類のP個のパイプラインの補間（内挿）ユニット103に連結される。メモリチップの数Mは、処理パイプラインPの数よりも少なくとも、それと同じでも、それより多くてもよいことが理解される。

【0066】上記の手段によって、メモリやバンクのコンフリクトによる遅延はなく、何れの観察方向からも1サイクル当たり1つのボクセルデータ値の一定したレートで、ボクセルメモリからデータを読み取ることが可能であるが、例外が1つある。その例外は上述したように、1つの行または列の終端のバンクが他の行または列の始点のバンクと同じ場合である。この例外が見つけられなければ、各DRAMチップは同じバンクから連続的な二番目のミニブロックを読み取るためにそのバンクをプリチャージする一方で、違反している列の終端では数サイクルの遅延が生じることになる。この遅延は図5のパイプライン全体に伝わり、余分な制御回路や複雑性を伴うこととなる。この問題を緩和するためには、図10に示したように、追加のパッファ244をDRAMチップ230と選択ユニット236との間に設ける。各パッファは、1つの列において単一のメモリモジュール230から読み取られる分のミニブロックを収容するには十分大きい。違反した列に遭遇すれば常に、DRAMチップからのフェッチングが、正常な上から下または左から右への方向ではなく、下から上または右から左の方向に行われ、また、アドレスジェネレータ102の制御下で進む。ミニブロックの列に関するデータはパッファに記憶され、行または列がそれぞれ正常な行または列である

か、違反している行または列であるかによって、先入れ先出しの順序または後入れ先出しの順序で選択ユニット236によって取り出される。

【0067】図示した実施の形態においては、ボリュームデータ集合はセクションに分轄されている。通常のセクションは横断方向の各列に32ボクセルまたは16ミニブロックを有する。メモリモジュールの数Mは4である。従って、各バッファ244は、列を何れかの順序で横断することができるように、4つのミニブロックのみを保持する必要がある。この手段によって、ボクセルデータは、観察方向に拘わらず、中断やアイドルサイクルの介入や遅延なく、DRAMメモリモジュールの最大定格速度のバーストで読み取ることができる。

【0068】以上、この発明のいくつかの実施の形態、並びにそれに対する修正や変更を説明してきたが、当業者には上記が単に解説的なものであって限定的ではなく、例示のみによって示されていることが明らかなはずである。多くの修正や他の実施の形態が当該技術分野の範囲にあり、請求項およびその均等物によって定義された発明の範囲に入るものと予想する。

【0069】

【発明の効果】以上のように、この発明によれば、1組のDRAMメモリモジュールに割り当てられたミニブロック内にボクセルデータが記憶されたボリュームレンダリングシステムのメモリサブシステム上に2層のスキューイングアーキテクチャを組み付け、それによって、DRAMメモリ最大のバーストレートでのデータ転送を可能にし、実時間ボリュームレンダリングを可能にすることができる。

【図面の簡単な説明】

【図1】 ボリュームデータ集合の線図である。

【図2】 光線キャスティングによって画像平面に投影されたボリュームデータ集合の画像の線図である。

【図3】 図2のボリュームデータ集合の断面図である。

【図4】 光線キャスティングによる個々の光線の処理の線図である。

【図5】 この発明による実時間ボリュームレンダリングのためのパイプライン化された処理エレメントのブロック図である。

【図6】 SDRAMへのミニブロックからなるボクセルのマッピングの線図である。

【図7】 メモリ内のミニブロックの線図である。

【図8】 DRAMのバンクと列の中のミニブロックの線図である。

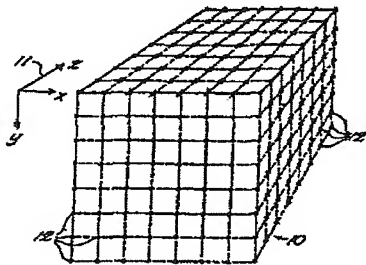
【図9】 ミニブロックのボクセルを自然な処理順序に再配置するためのデスキューイング回路のブロック図である。

【図10】 ミニブロックの列を読み取って何れかの順序で進行することを可能にする目的で、図9へのバッファの付加を示すブロック図である。

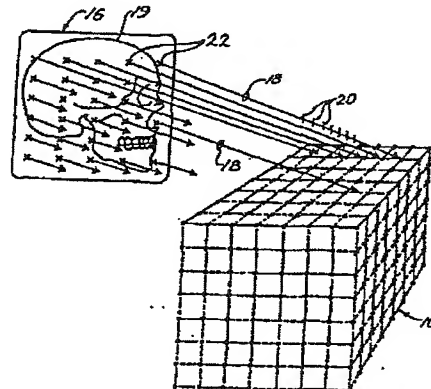
【符号の説明】

10 ボリュームデータ集合、11 軸、12 ボクセル、ボクセル位置、16 二次元画像平面、18 光線、19 画像、20 サンプルポイント、22 ピクセル、ピクセル位置、25 勾配、32 画像、100 ボクセルメモリ、110 FIFOエレメント、111 勾配推測ユニット、200 ミニブロック、202 ボクセル、204 3ビット二進数、206 軸、208メモリ、209 方向、211 方向、212 ミニブロック、214 グループ、230 DRAMチップ、234 メモリアドレス、236 選択ユニット、238 選択信号、240 ミニブロックデスキューイングユニット、242線、244 バッファ、300 破線、302 破線、304 破線。

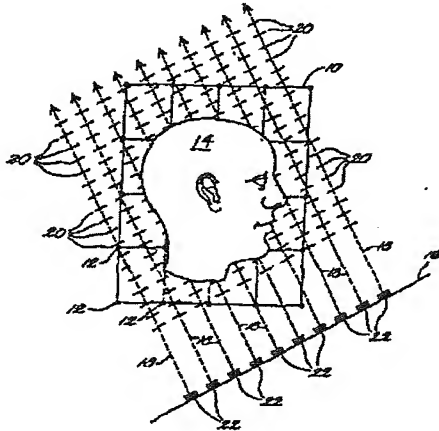
【図1】



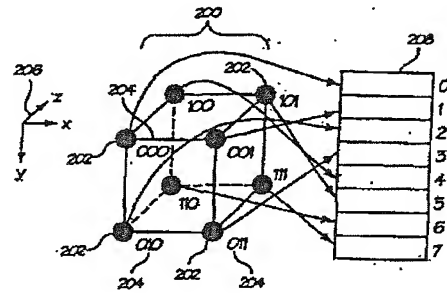
【図2】



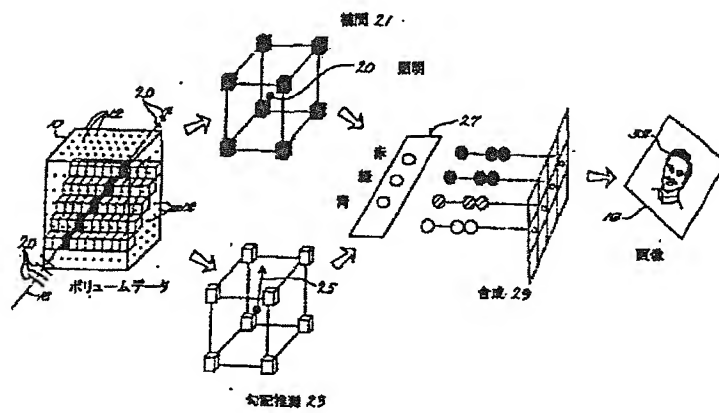
【図3】



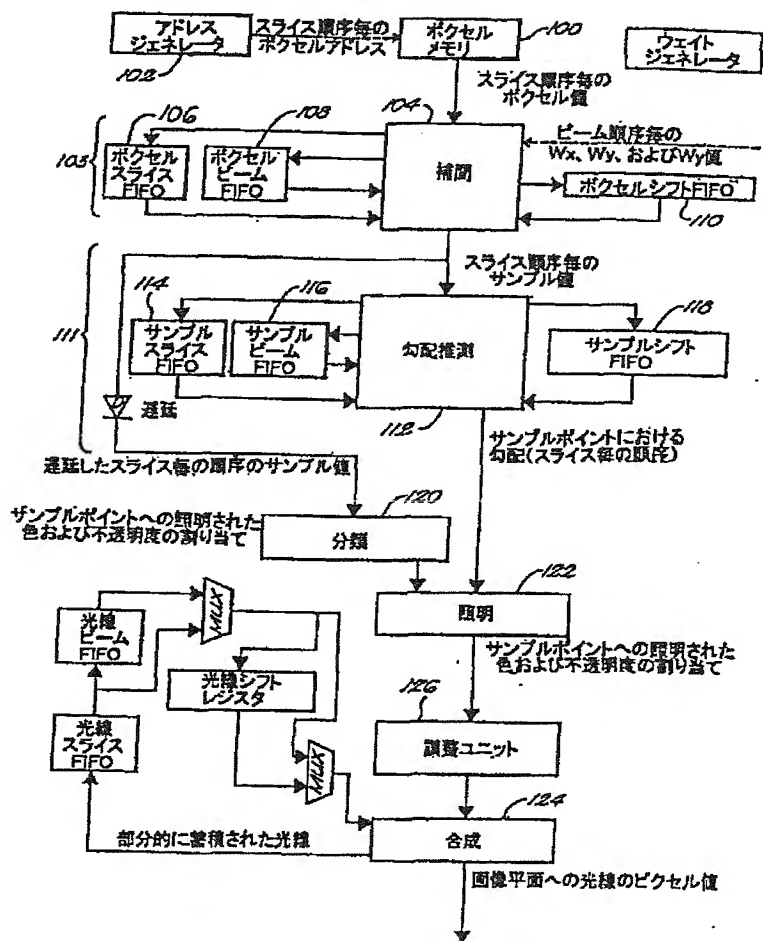
【図6】



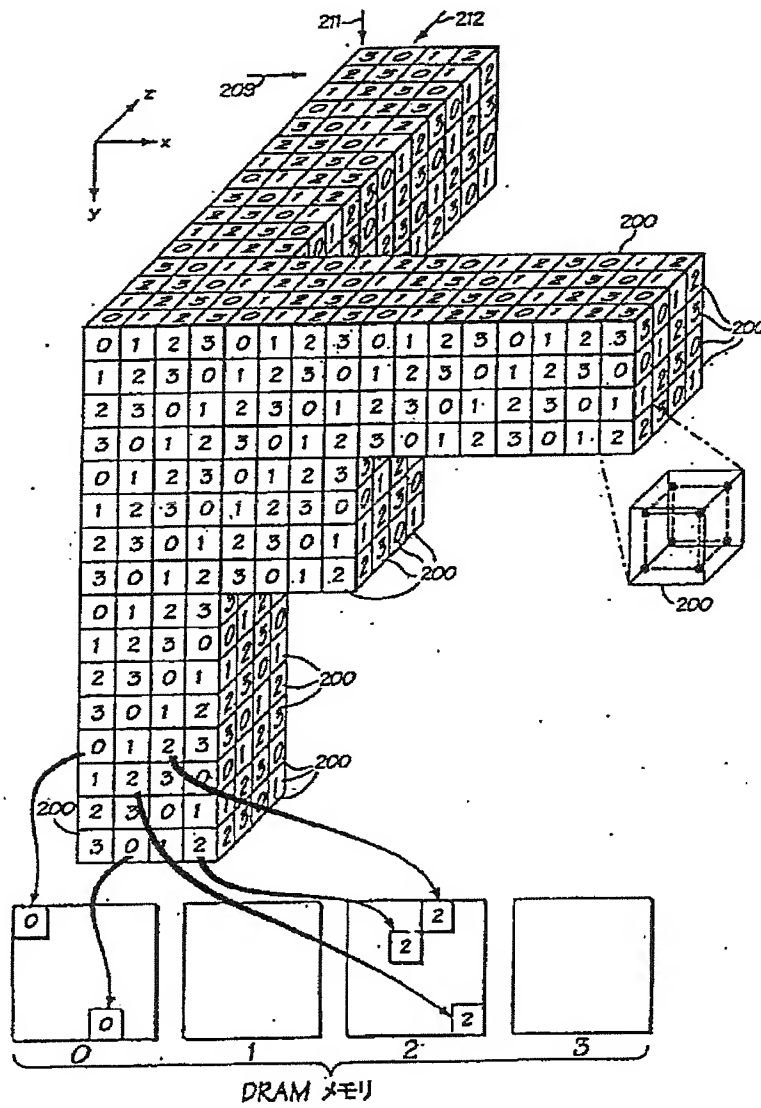
【図4】



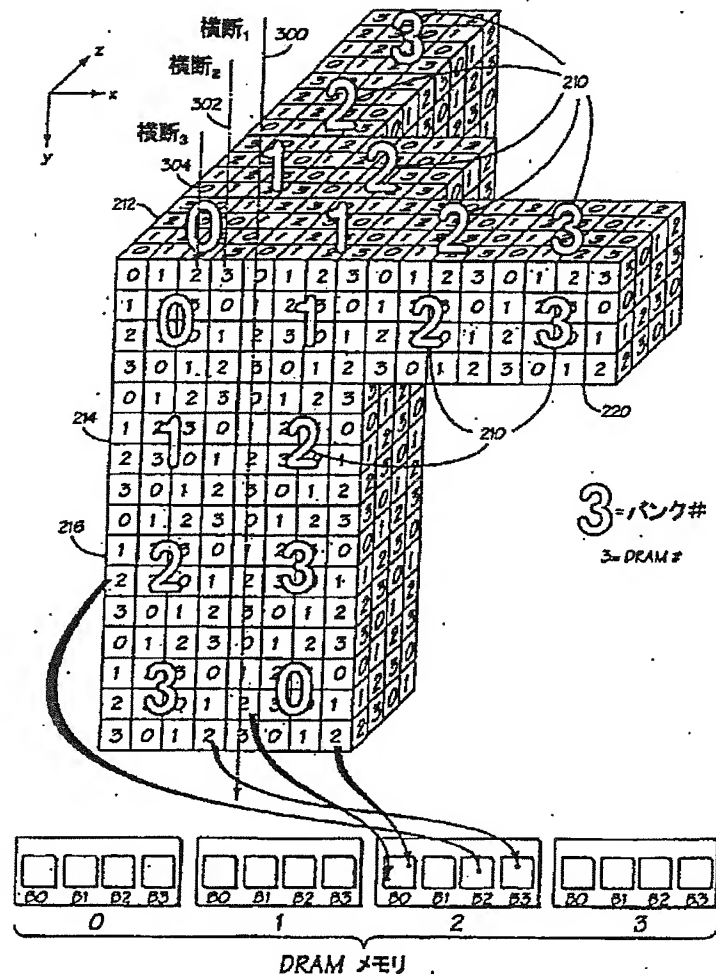
【図5】



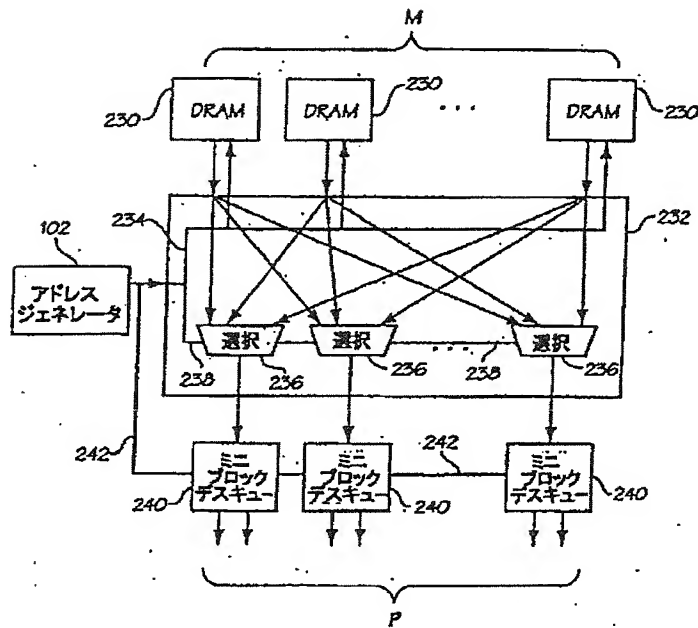
【図7】



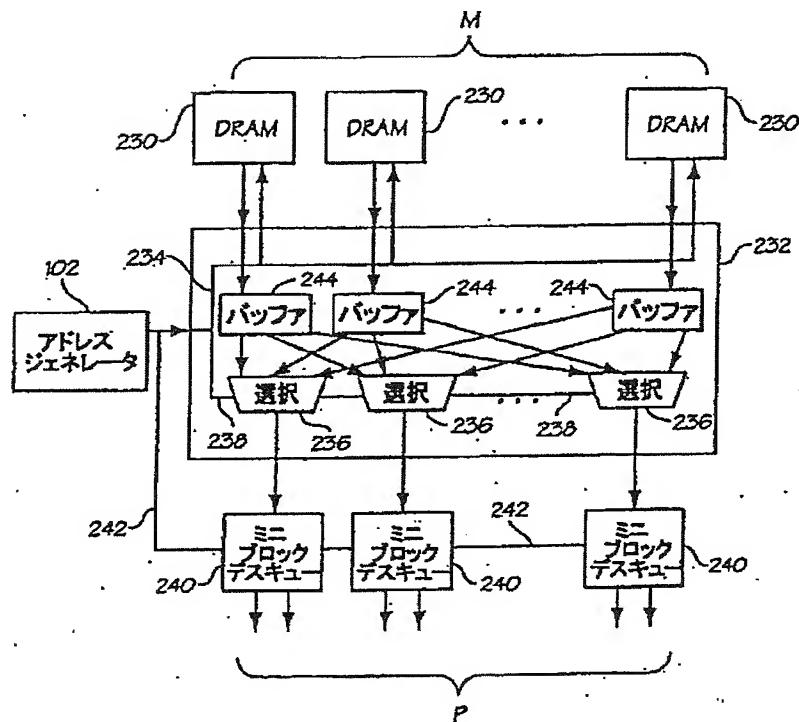
【図8】



【図9】



【図10】



フロントページの続き

(71)出願人 597067574

201 BROADWAY, CAMBRIDGE,
MASSACHUSETTS
02139, U. S. A.(72)発明者 ジェームズ・ニッテル
アメリカ合衆国、マサチューセッツ州、グ
ラトン、ヒル・ロード 241

(72)発明者 ウィリアム・ビート

アメリカ合衆国、マサチューセッツ州、ピ
レライカ、メリディエン・ウェイ 6

(72)発明者 ケネス・コレル

アメリカ合衆国、マサチューセッツ州、ラ
ンカスター、ルネンバーグ・ロード 2193